DEVELOPING ESH-17 SOFTWARE

Purpose

This Air Quality Group procedure describes the processes for developing ESH-17 software.

Scope

This procedure applies to the development of ESH-17 Category 1 software used to perform work for group projects and management. This procedure is recommended for development of Category 2 software.

In this procedure

This procedure addresses the following major topics:

Hazard Control Plan

The hazard evaluation associated with this work is documented in HCP-ESH-17-Office Work.

Signatures

Prepared by: _____ Steve Story, Information Management Team Leader	Date: 9/28/00
Approved by: _____ Terry Morgan, QA Officer	Date: 10/3/00
Work authorized by: _____ Doug Stavert, ESH-17 Group Leader	Date: 10/20/00

01/26/01

General information about this procedure

Attachments

This procedure has no attachments.

History of revision

This table lists the revision history and effective dates of this procedure.

Revision	Date	Description of Changes
0	11/1/00	New document.

Who requires training to this procedure?

The following personnel require training before implementing this procedure:

- Category 1 software developers
- Information Management Team Leader

Annual retraining is required and will be by self-study ("reading") training.

Training method

The training method for this procedure is "**self-study**" (**reading**) and is documented in accordance with the procedure for training (ESH-17-024).

General information, continued

Definitions specific to this procedure

Metadata: information stored in the database related to condition of collection and handling of data, such as who collected them, when, methodology, who prepared and reviewed the data, and who uploaded the data.

Structured Query Language (SQL): an ANSI industry-standard language used to manipulate information in a relational database.

Validation: A demonstration that a computer model (data and software) adequately describes physical reality over the range of variables of interest.

Verification: A demonstration that software correctly solves mathematical equations and performs the data processing it was designed to perform.

References

The following documents are referenced in this procedure:

- ESH-17-011, "Logbook Use and Control"
- ESH-17-024, "Personnel Training"
- ESH-17-025, "Records Management"
- ESH-17-034, "Network Server Backup Tape Rotation, Storage and Archiving"
- ESH-17-037, "ESH-17 Software Management"
- FMS, Inc., Total Access Analyzer for Microsoft Access: Database Documentation and Analysis for Microsoft Access, Version 8
- Litwin, Paul, Ken Getz and Mike Gilbert, Access 97 Developer's Handbook, Third Edition, Sybex.
- McFadden, Fred R., Jeffrey Hoffer and Mary Prescott, Modern Database Management, Fifth Edition, Addison Wesley.

Note

Actions specified within this procedure, unless preceded with "should" or "may," are to be considered mandatory guidance (i.e., "shall").

Developing category 1 software

Overview

Category 1 software is defined in procedure ESH-17-037 as 'critical software'. Critical software is database and/or application software developed by or for ESH-17 that stores data, performs calculations, or performs other significant operations on compliance data. Compliance data includes all data used to support compliance programs such as preliminary data, metadata, final data, and analytical data. Also included is data in any form such as hardcopy and electronic data. Software designated and used as Category 1 requires stringent levels of control to ensure data quality and legally defensible output.

To ensure data quality and legally defensible output, Category 1 software will be developed following the steps described in this procedure. These steps are:

- Project identification (follow ESH-17-037)
- Design
- Programming
- Testing (Validation and Verification)
- Maintenance

In addition to following these steps, the development of Category 1 software used by ESH-17 should be documented and the software application should have a user's manual. Documentation and manuals should be updated as new versions of critical software are developed. Critical software documentation, user's manuals, and version control logs will be maintained as ESH-17 records. Output will be identified to allow identification of the software version used to generate the output.

Design

Overview

Database applications should be designed before actual programming is started. Design includes:

- Identify the purpose of the software
- Determine what procedure(s) or process is being automated
- Identify required inputs and outputs
- Develop a process flow chart for the software
- Select software
- Validate the design process
- Document the software development plan

Purpose

Review the software description and purpose statement documented on the ESH-17-037 Software Category Form with the Project Leader and/or intended software users.

Procedure(s) or process

Determine what procedure or process is to be automated by the software. If there is no approved procedure, at a minimum, a documented process should be developed before continuing with this procedure.

Inputs and outputs

Determine and document the required inputs to and outputs from the software. When possible, use electronic inputs/outputs to avoid manual entry of data. Identify the requirements and format for reports (outputs).

Flow chart(s)

Using the procedure or documented process and the identified inputs/outputs, develop a flow chart. Review the flow chart with the Project Leader and/or software users before continuing.

Design, continued

Software selection

Make or Buy?
After reviewing the purpose and process, research the software market to determine if commercial software exists which will perform the process under review. If software does not exist, then proceed with the development process.

Development software
Commercial software used to develop ESH-17 applications must meet LANL Information Architecture Standards.

All ESH-17 database applications will be developed using Microsoft Access or SQL Server (unless approved by the Information Management Team Leader and the applicable Project Leader). Other software may be used to support the database applications. These include, but are not limited to:

- MS Office (Word, Excel, PowerPoint)
- Visual Basic
- Visual Basic for Applications (VBA)
- ColdFusion
- PERL (Practical Extraction and Report Language)
- HTML (Hypertext Markup Language)

Software version
Use the commercial software version currently in use by ESH-17.

Validation

Identify what processes, calculations and assumptions must be peer reviewed (validation) and include the results in your design documentation. The person performing the validation should be a subject matter expert other than the software developer.

Documentation –design plan

Write a short but concise software design plan which includes the items identified above. File this plan as a record in accordance with the last chapter of this procedure *Records Resulting from this Procedure.*

Programming

Overview

Programming should only begin after the design phase has been completed. Programming practices fall into two realms: those that must be implemented to maintain consistency within the group and those that are good practices but are of individual preference for each programmer.

Programmers must implement the following practices:

- Design for Y2K compliance
- Employ naming conventions
- Implement version control and tracking
- Develop a user's manual

Programmers should implement the following practices:

- Develop or use a common data dictionary
- Observe good programming practices with tables and queries
- Observe good programming practices when writing code
- Develop user interfaces when possible
- Document your software design

Each of these practices is discussed in the blocks below.

Y2K

Y2K compliant Date/Time fields must be enforced in database tables. The default date/time field in MS Access tables is not Y2K compliant. Set the following properties for each date/time field that you create in order to display all 4 digits of the year and to force input of a 4-digit year at data entry:

Format:	mm/dd/yyyy
Input Mask:	99/99/0000;0;_

If you set these properties BEFORE you develop any forms or reports based on the tables, then the forms and reports will inherit the properties automatically and you will not have to set them on the forms and reports. If you do not set the properties when you create the tables, and you have already created the forms and reports, then you must manually fix the forms and reports.

Programming, continued

Naming conventions

General:

- Use only letters, digits and underlines in table names and field names – no spaces or special characters. The first character must be a letter. Use the Caption property to display spaces or special characters in the field headings. If the Caption is assigned when the table is first created, it will propagate to all queries, forms, and reports subsequently based on that table.

  Example: Do not give fields names like "Site#", "Result (mg/ml)", or "%Uncertainty". Instead, name the fields "Site", "Result", and "Uncertainty" and set their respective Caption properties to "Site #", "Result (mg/ml)", and "%Uncertainty".

- Use 30 characters or less

Code Variables:

- Whenever feasible, prefix variables in VBA code with Hungarian notation that indicates the data type of the variable. Refer to the "Access 97 Developers Handbook."

  Examples: intRecordID, dblResult, strFirstName

Object Descriptions

- Include a short (≤ 255 characters) description of each database object in the object's property sheet. Also fill in the description property of each table field.
- Give all objects concise, yet descriptive names. Avoid single-character variable names in code, except as loop counters where their usage is obvious (for n = 1 to 5 do ….). Also avoid overly verbose object names that are cumbersome to use in code and subject to typographical errors.
- When the same field appears in multiple tables, make sure the name, data type, and size are consistent.

Programming, continued

Version control and tracking

Version control will be used with all category 1 software.

- An acceptable method of version control is to use an identifier, such as a version number or date stamp, on all output. This identifier must be obvious to the user and tie the product to the software version used. Then, maintain a log (according to ESH-17-011) of the inclusive dates each critical software version was used. Submit this log to the records coordinator along with the software documentation and user's manual for each version.
- Other methods may be acceptable with prior approval from the Information Management Team Leader.

User's manual

Develop a user's manual for Category I software before releasing it for production. User's manuals must include:

- the purpose of the software;
- reference to the procedure or documented process that the software automates or fulfills the requirements of;
- instructions for using the software,
- a flow chart of how the software functions, if needed.

Clearly state to the user (in the software documentation or users' manual) the principles, theory, or purpose of the software, the method of verification used, and the range or limits of verification.

Data dictionary

Develop a common data dictionary for your database. If a group common data dictionary is available, use it. Use appropriate naming conventions for your dictionary.

Programming, continued

Tables/queries

- Use primary keys and indexes
- Eliminate cut and paste operations in databases to transfer data
- Whenever possible, reuse queries and prompt users for required criteria instead of creating multiple, almost-identical, queries that perform essentially the same function but with slightly different criteria: e.g., instead of creating "1997ResultsSummary", "1998ResultsSummary", and "1999ResultsSummary" … create one "ResultsSummary" query and prompt for the year or date range.
- Complex mathematical calculations, string parsing, and manipulation, and other intricate operations should usually be coded as functions in an Access module, especially if they are needed in multiple places throughout the application, because functions in modules can be called from anywhere in the database, and the functionality can be documented in the comments.
- Avoid performing complicated operations "on the fly" in the query designer – this practice is more prone to typographical errors, makes the calculations difficult to document and verify, complicates troubleshooting, and requires duplication of effort to retype the calculations each time it is used.

Code

- Comment VBA code liberally to ensure that another programmer (or you, six months later!) can understand the functionality of the code. Comments should include at minimum a brief paragraph at the top of the procedure summarizing the purpose of the code, the programmer's name, date of creation and date of last modification. This summary should explain any mathematical equations performed by the module and define any variables and constants used in the equation. Reasons for significant code modifications after the code goes into production should be added to the summary as well. Shorter, one or two line comments should be used in the body of the code as needed to explain the functionality where it is not obvious.
- Use "Option Explicit" to require declaration of all variables.
- Assign a data type to the value returned by functions.

  Examples: Function Calculation(dblInput) as double.
  Function ConcatenateStrings(strFirst, strSecond) as string.
- Use appropriate line indentation in your code to clearly delimit control structures – especially when they are nested. This makes the code easier to read, understand, and debug.

Programming, continued

User interfaces

Provide user interfaces, such as input forms, as much as possible. Include data validation during the input process.

Document design

Develop a software design document that defines how the application is constructed and how the components interact. This document should be comprehensive enough that the original developer or another developer can use it as a reference for future application modification. Commercial tools are available to automate database analysis and documentation. One such product, Total Access Analyzer by FMS, Inc., can generate over 225 types of reports. It is not necessary to print all 225 different reports for every application developed! Developers using this or any other analysis tool on a database application should preview the reports available and print out a set of reports that are appropriate to document that particular application. Using FMS Total Access Analyzer, an example set of basic database documentation might include the following reports:

- Application Diagram: Graphical diagram of your application and procedure flow. See the entire hierarchy of how your entire application flows across procedures, forms, events, macros, etc.
- Data Diagram: Graphical diagram of how your data flows. Discover the entire family of objects based on a table. See the entire hierarchy of how tables flow into queries which flow into other queries, forms, and reports.
- Object Diagram: Graphical diagram of what objects are referenced by a form or report. Understand all the objects a form or report is based upon including tables, queries, macros, and other forms and reports.
- Database References: List of all VBA references in the database.
- Table Dictionary: Detailed information for each table with its table properties, indexes, and fields.
- Table Fields: Each documented table with its fields including field types and descriptions.
- Table References: For each table, where it is referenced from.
- Query List with SQL: List of documented queries with its description and SQL property.
- Query References: For each query, where it is referenced from.

Programming, continued

Document design, *continued*

- Form Controls: For each form, its controls sorted alphabetically
- Report Dictionary: Comprehensive information on each report with its properties, sub-reports, sections, and controls.
- Macro Dictionary: Each macro and its commands.
- Bracketed Module Printout: Module code printout with standardized indentations and brackets around programming loops.

This report list is merely a suggestion. Developers can include or substitute additional FMS reports, or manually-created narrative, as they deem necessary. Screen captures of forms, sample pages of reports, and detail design flow charts are other examples of documentation that might be useful to include in the software design document.

Testing (validation and verification)

Overview

Before using Category 1 software in production (new version), the software must be tested (validated and verified) to prove that the software operates as required and the process must be documented.

Validation

Validation of the design process should have occurred during the initial design phase. Review the initial validation documentation and the software application to determine if any changes occurred during programming that should be peer reviewed at this time before verification tests are performed.

Verification

Determine what tests must be run to prove the software performs as required (verification)

- Develop a test plan and a set of test input data or instructions that simulate inputs at the upper and lower limits of expected data or instruction ranges.
- Determine the expected output using manual calculations or methods, or other sources of known results.
- Run the tests and compare the actual results to the expected results. If problems are found, make corrections or changes and re-test the software until the expected results are obtained.

Documentation – Validation / Verification

Document the test process and results. Include:

- The software version
- Data or instructions used for testing
- Test results
- Name and signature of person who performed the test
- Test date
- Peer review of test

Maintenance

Overview

All ESH-17 Category 1 production software applications reside on a server database drive which is backed up on a regular interval. Read and write access must be controlled to ensure integrity. Production software is used until a revised version is produced and approved for use.

Location

All Category 1 databases reside on the ESH-17 network server on the databases drive in the appropriate project directory.

Security

The **ESH-17 Database Administrator** assigns permissions to database users using Microsoft Access Security.

Backup

Databases located on the network server on the databases drive are backed up daily in accordance with ESH-17-034, "Network Server Backup Tape Rotation, Storage and Archiving." A backup copy of the source code and executable code in machine-readable form should be kept in a remote location as a final form of backup.

When is a version change required?

A change in version designator is required after any "non-cosmetic" change is made to the structure or function of a table, query, form, report, macro, or module. The interpretation of what constitutes a "non-cosmetic" change will be made by the software developer and the responsible project leader.

Some steps of the software development cycle should be completed to document changes to the software. Notify software users that a new version is available.

Records resulting from this procedure

Records

The following records generated as a result of this procedure are to be submitted as records **within 4 weeks** of version release to the records coordinator:

- Software design plan
- Test plan and results
- Users manual
- Version control log, if used
- Software design documentation (optional)